Margin Notes:

Building a Contextually Aware Associative Memory

Bradley J. Rhodes
MIT Media Lab
20 Ames Street
Cambridge, MA 02139 USA
+1 617 253 9601
rhodes@media.mit.edu

ABSTRACT

Both the Human Computer Interaction and Information Retrieval fields have developed techniques to allow a searcher to find the information they seek quickly. However, these techniques are designed to augment one's direct-recall memory, where the searcher is actively trying to find information. Associative memory, in contrast, happens automatically and continuously, triggering memories that relate to the observed world. This paper presents design techniques and heuristics for building "remembrance agents," applications that watch a user's context and proactively suggest information that may be of use. General design issues are discussed and illuminated by a description of Margin Notes, an automatic just-in-time information system for the Web.

Keywords

Contextual interfaces, software agents, remembrance agents, World Wide Web, browsers

1. INTRODUCTION

When using a tool or solving a problem, people will use a combination of their own knowledge and knowledge inherent in the environment. Through the use of affordances and other design techniques, a good design will endow the environment with as much information as possible to aid a user in solving their problem. However, no design can provide information for every situation, and no designer can include personalized information for every user. For example, a designer of a web page might add a hyperlink defining difficult terms or concepts in a project proposal, but they cannot add links letting a particular reader know that this proposal is similar to a project being implemented by the reader's own workgroup.

One solution to this problem is to create a "remembrance agent" (RA) that continuously watches a person's environment and provides information relevant to that person and their current situation. The provided information might be from a task-specific database, such as an address database, or from a more free-form information source such as personal email archives. For example, an early form of the remembrance agent software [9] automatically reminded a user of personal email and note files relevant to what was currently being typed into a word processor. Depending on what information is provided, an RA can act as a personal memory system, an organizational memory or a general just-in-time information system. [1]

Remembrance agents are similar to search engines, notification software, and personalized newspapers in that all provide personal information to a user, but there are also important differences. Examining these similarities and differences can inform the creation of design techniques for building RAs.

Like search engines, RAs provide information from a rich database given fuzzy and incomplete knowledge about what information should be presented. However, search engines and structured knowledge bases such as Yahoo! are inherently interactive: an information-seeker has some query in mind and directly interacts with the program to obtain the desired information. Remembrance agents, on the other hand, are proactive. The user need not have a query in mind, or even know that information relevant to her situation exists. An RA gives information without any prompting, so users need not know that useful information exists in their given context. However, receivers of this unrequested information may be less willing to take the time to process it, and are less willing to be distracted from their chosen task.

Notification software, like RAs, is proactive. Most of the time an alarm sits in the background and doesn't interrupt its user. When a new piece of email arrives, a stock price goes below a certain threshold, or another trigger event occurs, the alarm activates and informs the user. Both RAs and alarms also have the potential to be context-aware, using the user's own context to trigger information. However, RAs are designed to provide much richer and more varied kinds of information than alarms. Alarms usually provide only a single piece of information, namely whether the alarm's trigger event has occurred. Even if the algorithm to decide whether to trigger an alarm is very complex, the end result is by definition only one of a very limited set of states. On the other hand, a single RA might provide information from any of

IUI 2000 New Orleans LA USA

[1] RAs are "software agents" in that they watch an environment and can act based on that environment without direct user intervention. They should not be confused with what are often called user interface agents or with distributed agent architectures.

thousands of documents, emails, or database entries with each "suggestion" leading to pages of information.

From a design standpoint, the most important difference between notification software and RAs is that a user is expected to know what an alarm means when she sees it. In some cases that knowledge is trivial. For example, it is obvious that you have new email when a letter appears in the iconic representation of your mailbox. In other cases that knowledge is the result of years of training, such as the knowledge required for a pilot to know what it means when a "oil pressure low" warning light flashes. Because RAs can show a variety of different pieces of information, the user cannot be expected to know what a particular suggestion contains. The display interface for an RA must be able to present that information.

Finally, RAs are similar to personalized newsfeeds that automatically send email when news that fits a personal profile is announced. Both systems are personalized and proactive, and both provide rich information from a large source pool. However, personalized newspapers aren't related to the receiver's current context. Instead, the personalization comes from manually or automatically generated user profile, while the trigger event to send a news article is usually far outside the receiver's personal environment. For example, a personal news system might watch the AP newswire and automatically compare articles with a subscriber's personal profile. When a close match is found, the news article is automatically sent to the subscriber's email. In some systems, the headline and first paragraph are also sent to the user's alphanumeric pager. However, the "sensors" in these systems are all looking at the newswire, not at the user. Personalization comes from a static user profile. In RAs the situation is reversed. The sensors of the system watch the user and the user's current context, and the personalization comes from these dynamic user-oriented sensors. Additional personalization occurs when drawing suggestions from a personal information database such as email archives.

The next section will introduce the Margin Notes system, a remembrance agent that on the fly annotates web pages with relevant information from a user's own email or other text files. The system has been in operation for two years as iterative user testing has been performed. Evaluations of the system are presented, and the system will then be used to illuminate design techniques for creating remembrance agent software.

2. MARGIN NOTES

2.1 Overview

Margin Notes is a remembrance agent that automatically rewrites web pages as they are loaded, adding hyperlinks to personal files. As a web page is loaded, Margin Notes adds a black margin strip to the right of the document. It then compares each section of the document to pre-indexed email archives, notes files, and other text files, based on keyword co-occurrence. If one of these indexed files is found to be relevant to the current section of the web page, a small "suggestion box" is included in the margin next to the relevant section. The box contains a quick description of the suggested text, an bar representing the relevance of the suggestion, and a link to get more information. The suggestion box usually consists of a subject, date, author, and filename for the suggested text.

Placing the cursor over the suggestion box produces a list of the five keywords that were most important in deciding a suggestion's relevance to the annotated web page section. This list appears both in the message area at the bottom of the browser and as a pull-down window under the box itself.

Clicking on a suggestion box jumps to the full text of the email, note-file, or text being suggested. The suggested page also has feedback buttons used for evaluating the system.

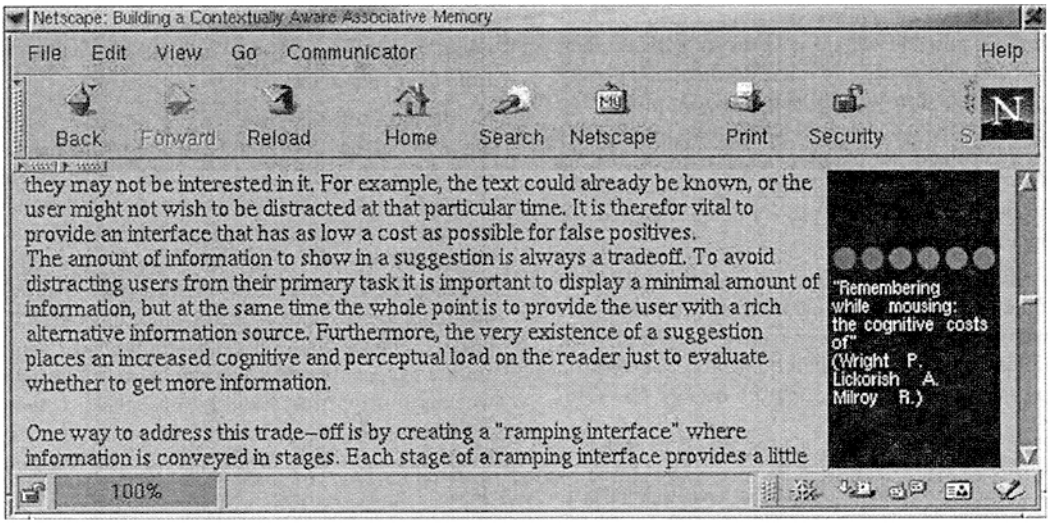

Figure 1: A screenshot of the web version of this paper, annotated by Margin Notes. The database used for this example is a subset of the INSPEC database of paper abstracts. The suggested document, listed in the black margin to the right, is a paper from SigCHI Bulletin that discusses the display of information and cognitive load. The suggested document was previously unknown by the author, but is now being added to his thesis reading list.

2.2 Internals

Before running Margin Notes, potentially relevant documents are indexed to allow quick retrieval. Documents can be from a generally useful database such as a collection of newspaper or journal articles, or from personal sources such as email or notes. Pages can also be indexed directly from the web. Documents are re-indexed nightly to incorporate new changes and additions.

The system is a proxy server, and acts as an intermediary between the Web-browser and the Web. Previous versions of the system annotated on the fly, which required that the browser not show sections of the requested web page until the annotation was computed. This delay proved unacceptable to test users, especially for long documents. In the new system, a page is immediately passed on to the browser with the full text intact, but with the addition of a black margin to the right of the text. As suggestions are computed they appear in the black margin next to the section they annotate.

After sending the document to the browser, the text of the document is broken into sections based on header, paragraph, list, horizontal rule and center HTML tags. Small sections are ignored, as are JavaScript, comments and HTML tags. The remaining sections are passed to a relevance engine called Savant, which uses text-retrieval techniques to find documents relevant to the current section.

Savant looks at each document in the indexed database and computes a relevance score relative to the processed section of the web page. This score is based on co-occurrence of words using a term frequency / inverse-document frequency (TFiDF) method [11] and the Okapi weighting scheme [12]. Savant returns the document number for the five most relevant documents, along with their relevance score and a brief summary-line. Savant also compares title and header fields in the HTML to the equivalent fields in the returned document. For example, the header for the section of the web page being visited will be compared to the title of journal articles in the INSPEC database, and this relevance will be averaged into the overall score for the suggestion.

In previous versions of the system, users complained that while suggestions were often related to the section they annotated, no suggestion related to the big picture. To solve this problem, the first annotation on the page is now calculated using the entire web document as a query, with the first section is weighted more highly than the rest of the document. The remaining annotations are more focused and still use a single section as their query.

Of the documents suggested by Savant, the highest relevance document that has not already been suggested for this web page is selected. If the relevance score is above a threshold a suggestion-box is created and inserted into the margin of the web page at the top of the relevant section. The relevance threshold can be changed depending on the source database, and is adjustable for individual users.

When a suggestion link is followed, a script is run that retrieves the document and sends it to the web browser. The document might be reformatted based on its type; for example email headers are printed in a different font. For evaluation purposes, the script also logs the fact that a suggestion was followed, and inserts buttons at the top of the screen allowing the user to rate the suggestion.

3. DESIGN ISSUES FOR CONTEXTUALLY RELEVANT JUST-IN-TIME INFORMATION

The most important design constraint for remembrance agents is that reading suggestions must be a secondary task for a user. Unlike users of a search engine, users of remembrance agents are not actively seeking the information being suggested. They are therefor less tolerant of being distracted from their primary, chosen task, and are less willing to dig through suggestions to get useful information. Furthermore, even if a suggested text is highly relevant to the user's current context they may not be interested in it. For example, the text could already be known, or the user might not wish to be distracted at that particular time. It is therefor vital to provide an interface that has as low a cost as possible for false positives.

The amount of information to show in a suggestion is always a tradeoff. To avoid distracting users from their primary task it is important to display a minimal amount of information, but at the same time the whole point is to provide the user with a rich alternative information source. Furthermore, the very existence of a suggestion places an increased cognitive and perceptual load on the reader just to evaluate whether to get more information.

One way to address this trade-off is by creating a "ramping interface" where information is conveyed in stages. Each stage of a ramping interface provides a little more information, at the cost of the user spending a little more of their attention to read and understand it. The idea is to get useful information to a user quickly, while at the same time allowing them to bail out on an unwanted suggestion with as little distraction as possible.

In a ramping interface, the user should always be able to get more information by going to the next stage, and the effort required to get to that stage should be proportional to the amount of information provided in the current stage. It should only require a simple action, such as moving the mouse to a target, for a user to go to early stages. Going to a later stage might require the user to pick a topic from a menu, trading off simplicity for increased control of what information is displayed.

For example, in the Margin Notes system the first stage is what happens when the system analyzes a section of a web page and decides not to leave any suggestion at all. At this stage, the input is simply a passive sensor watching the user's actions. No user action or attention is required to show the agent what to do. The output at this stage is nothing: the agent simply passes the HTML to the browser and continues.

When Margin Notes determines that a suggestion is beyond a certain relevance threshold, it automatically jumps to the second stage and shows the suggestion box. In keeping with the philosophy that the effort in jumping to the next stage should be commensurate with the amount of work or attention required by the current stage, no user thought or action is required to display the suggestion box.

At this point the user may ignore the suggestion entirely, and the interaction has cost nothing more than some screen real estate and a very minor cognitive load. If he wishes to go to the next stage, he can quickly view a graphical bar of filled-in circles indicating the relevance value for that suggestion. In earlier versions a two-digit number was used to indicate relevance. While the same information was available, it was much more difficult to find and

process quickly. The relevance bar, on the other hand, only requires perceptual processing to determine the relevance of a suggestion.

The next stage is reached by reading the suggestion description, which requires further attention on the part of the user. Information in the suggestion box is as terse as possible to allow rapid scanning for content. The box also contains many different kinds of information to try to contextualize the suggestion. For example, when email is displayed the box contains subject, author, date and archive filename in the hopes that at least one of these will give a good indication of the suggestion's content.

At the fourth stage of the Margin Notes interface the system displays the most relevant keywords. Going to this stage requires physical action by the user (a mouse-over of the link), and gives an incremental increase in the amount of information returned. While keyword information could have been included in the original suggestion (reducing the system to a four-stage ramping interface) it was decided that this information made the suggestion-box too cluttered. To jump to the final stage, the user clicks on the link and gets the fully suggested text. At this point the user is completely committed to seeing the text, and has been distracted from their primary task. However, even at this stage the text is displayed in a separate window, allowing the user to keep their initial context on the screen. Furthermore, the suggestion is still at least related to the user's original context. Hopefully if the user got to this point the suggested text was worth the attention spent.

It should also be noted that a user will automatically visit each stage of the ramping interface through normal interaction. In the case of the Web, the natural action to get more information is to click on a link. This requires first seeing the link in one's peripheral vision, then seeing it in the fovea, then moving the mouse over the link and clicking. At each stage of this set of actions the ramping interface gives a little more information, allowing the user to bail out if he chooses.

When designing a ramping interface it is helpful to consider at what stage a user is likely to receive the information they need. In the Margin Notes system, it is assumed that most of the time users will receive information from the full suggested text, but that occasionally they will be reminded by the suggestion box itself and never need to follow the link. On the other hand, remembrance agents designed for a car or other attention-demanding environment might be designed such that users normally needn't go past the first stage of a suggestion.

Another important design consideration is how to indicate to which part of a user's context a suggestion is relevant. Even if the user's full context is constrained to a single web page, it still needs to be apparent whether a suggestion is relevant to a single paragraph, a section, or the entire web page. This indication can be by fiat, e.g. by simply declaring that all suggestions are chosen based on relevance to the whole body, or by indicating the relevance somehow in the suggestion. In the Margin Notes system, suggestions are defined to be relevant to the section next to which they appear, within the context of the whole page.

Finally, it is important to be able to distinguish a suggestion from the user's other context. In some interfaces this is trivial, either because suggestions always appear out-of-band or because suggestions look nothing like anything that would normally be seen. Augmented reality systems, for example, fall into the latter category because the quality of computer screens is not high enough to be mistaken as a part of the real world. This is not the case for the Margin Notes system, where HTML is actually modified before being shown. In this case, suggestions need to be "branded" in such a way that they are clearly distinguished from the web page being viewed. This branding is accomplished by placing all Margin Notes within its own black border, producing a natural split between the web document's space and the Margin Notes space.

Equally important is what information to show, or indeed whether to show any information at all. In some applications, what to show is relatively trivial. For example, the Emacs Big-Brother Database automatically displays the contact information for the sender of any email received. Because the program is limited in scope, what to show is determined by a simple database lookup.

The task of finding relevant information is much harder when the information being suggested is richer than a simple database. In remembrance agents where both the context and suggested information is text, many techniques can be borrowed from the Information Retrieval community. However, selecting what information to show is more difficult than returning best matches to a search-engine query for several reasons.

The first complication is that remembrance agents need to not only return a best match, but must also determine whether to report a suggestion at all. This task is a combination of information filtering and information retrieval, and requires modification to many of the standard search-engine techniques. Because users are less willing to search through multiple suggestions for a good hit, it is also important that the search technique used be very precise, with a high likelihood of getting a good hit in the first few results returned.

Another potential complication is that remembrance agents are not basing their suggestions on actual user-generated queries, but only on passively detected user context. This throws all available information into a query, without the user being able to specify useful content first. Similarly, it can be difficult to detect context shifts, so suggestions may refer to something in which the user is no longer has an interest. On the other hand, the system shouldn't focus to narrowly on the exact context at hand or it will never see the forest for the trees. Margin Notes compromises by performing query expansion based on relevant keywords in the overall document, but still annotating individual sections of the web page.

Finally, the value of a remembrance agent is highly dependent on the source of suggestions. First, it is much easier to provide terse descriptions of a suggested text for personal information than non-personal, simply because the user has probably already seen the text before and knows the people and events involved. News and technical articles are less easy to contextualize, but have still been written for a general audience with a known general background. News and technical articles also have the advantage that their titles usually are written to convey the general idea of the full text. The hardest kind of information to convey is non-personal random information like web pages, where a title such as "Bob's Home Page" gives little information as to the page's actual content. Second, suggestions from high-quality, personally relevant corpora of information naturally tend to produce better suggestions. If the database is a poor match then at best no suggestions will be shown, at worst irrelevant and useless suggestions will be displayed. While this is equally true for search

engines, a bad suggestion for a search engine is far less of an annoyance than for a remembrance agent. As was described by one beta-tester, when a search engine returns bad hits he just tries different search terms or another engine. When Margin Notes produces a bad suggestion he is annoyed that it cluttered his screen without merit.

4. EVALUATION

The usefulness of Margin Notes, and remembrance agents in general, depends greatly on how well their database meshes with user contexts and users themselves. Given a narrowly constrained task domain it is often possible to find a corpus containing only high-quality information. However, the ultimate goal of this research is to produce a system that will provide useful information in everyday situations, given data that is readily available. Task-performance analysis is still possible within these constraints, but it becomes necessary to show both that the tasks presented are typical of real-world tasks, and further that the database used is general enough that it will be useful for a wide range of tasks.

It is also difficult to evaluate Margin Notes based completely on the relevance of suggestions it provides, as might be done for an information-retrieval system. The problem is that relevance (or more accurately, "precision" and "recall") do not necessarily indicate usefulness of a suggestion. For example, a suggestion might be extremely relevant, but if the user has already seen the text, or if the suggested text is incorrect, or if the user just doesn't want to be distracted then the suggestion is not useful. As a real world example, Margin Notes will occasionally report a suggestion with an extremely high relevance. In some of these cases, the text is an exact copy of the web page being viewed, with no extra information at all. Certainly relevance is still a heuristic for usefulness, but usefulness should also be evaluated on it's own.

Margin Notes has been used by up to five people at a time for over two years, and a previous version was tested with long-term twelve users. Of the testers, three ran personal copies of Margin Notes that pulled suggestions from their own email archives, four ran personal copies that pulled suggestions from all their computer files (including email archives), and five ran a generic copy that pulled suggestions from the lab-wide email archive. The personal databases ranged from 762 unique files to almost 30,000 unique files. The generic database was the largest with over 115,000 emails ranging back to 1986. After the month-long test, the beta-testers were asked to fill out surveys giving their opinions on the system. Eight of the twelve original beta-testers returned surveys. Logging data tracked how many suggestions were displayed per page and which suggestions were followed. Results from this evaluation were mixed, and many of the problems that were uncovered have been addressed in the new version.

Of the 4576 suggestions shown during the period in question, only 6% of them were followed to the full-text. The most likely explanations, based on interviews of the users, are that users found most suggestions to be irrelevant or useless to their current situation, or never read the suggestion note to start with. Indeed, when asked to rank suggestion quality, speed of loading, not cluttering web pages with lots of suggestions, and making sure to show a relevant suggestion if one exists, all but one of the eight respondents said the most needed improvement for the current Margin Notes system was suggestion quality. When asked to rank the same four features in terms of importance for a Margin Notes system in general, all ranked suggestion quality in the top two (five as most important, three respondents ranking it second). This result has since lead to a complete rewrite of the Savant search engine with a new algorithm that uses more information from the current web page and indexed documents.

Users were also asked to rate on a five-point scale how often the suggestion box was a useful reminder in itself, such that they needn't look at the full-suggested text. With five being "very often," users gave an average answer of 3.17 (standard dev = 1.33).

However, in spite of the poor click-through on suggestions, overall user response was positive. When asked to rate on a five-point scale how likely they were to use the system after the experiment was over (five being very likely), the average result was 3.63 (standard dev = 0.74), with none of the eight respondents giving less than a three.

These evaluations have motivated further changes to the system, and new evaluations are being conducted to see the effects of these improvements.

5. RELATED WORK

Margin Notes is most related to other systems that automatically annotate documents with hyperlinks to more information. One such system is the Peace, Love, and Understanding Machine (PLUM) system [3]. PLUM will automatically add hyperlinks to disaster news stories to better give a reader a personalized understanding of the news. Another is VOIR [4], which combines elements of a document to be annotated with a user's interest profile to automatically create hyperlinks. A third is Watson [1], which automatically produces and submits an Alta-Vista query based on a user's current web page or Microsoft Word document. Results are clustered and displayed in a separate window.

Also similar are systems that automatically link to recommended documents based on a user profile. One such system is Letizia [7]. Letizia automatically creates a user profile by compiling keywords contained in previous web pages viewed by the user. It then acts as an "advanced scout," following links from the user's current and subsequent web pages and bringing up pages that match that profile. Letizia is in many ways the reverse of Margin Notes. Margin Notes generates "queries" based on a user's current web page, and applies this query to a relatively static database of potentially relevant documents. Letizia, on the other hand, generates queries based on a slowly changing personal profile built up from past browsing habits. This query is then applied to the set of documents linked off of the web page currently being visited. Another similar system is WebWatcher [5], which highlights existing hyperlinks that best match a user's stated interest. Like with Letizia, recommendations are only pulled from links off the page currently being viewed, based on a user profile. Recommended links are highlighted with a special icon.

Margin Notes also shares interface features with new methods for displaying hyperlinks. For example, the XLibris pen-based system [8] supports the ability to hand-annotate documents using margin links. Similarly, the Fluid Links system [13] allows web designers to create "glosses," which are a way of implementing a ramping interface.

Margin Notes is also similar to the Forget-Me-Not system [2][6], which is also designed to organize personal information without

user intervention. However, Forget-Me-Not is designed primarily as a direct memory aid, allowing users to search a database of their own automatically recorded past actions. It falls into the same category as search engines and other interactive information retrieval systems, in that it is not designed to be proactive.

Finally, the Margin Notes work was based on previous work examining remembrance agents in word-processors [9] and on wearable computers [10].

6. FUTURE WORK

Based on the initial feedback, the next step is to improve the quality of suggestions produced by the Margin Notes system. Currently, short sections in web pages are more likely to produce hits than large sections, because there are fewer keywords that have to match. The next version will fix this problem. Finally, the number of beta-testers will be extended to get more data points for analysis, using both task-performance analysis and logging of long-time usage.

7. ACKNOWLEDGEMENTS

I'd like to thank my sponsors at British Telecom and Merrill Lynch, and Jesse Rabek, Jesse Koontz, and Jan Nelson, without whom this project would never have gotten off the ground. Finally, I'd like to thank Rich Devaul and Alan Wexelblat for their numerous suggestions and edits.

8. REFERENCES

[1] Budzik, J., and Hammond, K. Watson: Anticipating and Contextualizing Information Needs. In *Proceedings of the Sixty-second Annual Meeting of the American Society for Information Science.* 1999.

[2] Brown, P., Bovey, J., and Chen, X. Context-aware applications: from the laboratory to the marketplace. IEEE Personal Communications, 4(5), October 1997.

[3] Elo, S. PLUM: Contextualizing News for Communities Through Augmentation. Thesis for Master of Science in Media Arts and Sciences, MIT, 1995.

[4] Golovchinsky, G., What the Query Told the Link: The Integration of Hypertext and Information Retrieval, in The Proceedings of HyperText 97, Southampton UK, 1997.

[5] Joachims, T., Freitag, D. and Mitchell, T. WebWatcher: A Tour Guide for the World Wide Web, in Proceedings of IJCAI97, August 1997.

[6] Lamming, M., Brown, P., Carter, K., Eldridge, M., Flynn, M., Louie, G., Robinson, P., and Sellen, A. The design of a human memory prosthesis. The Computer Journal, 37(3), 153-163. 1994.

[7] Lieberman, H. Autonomous Interface Agents, in Proceedings of CHI-97, Atlanta, Georgia, March 1997, ACM Press, 67-74.

[8] Price, M., Golovchinsky, G., Schilit, B., Linking by Inking: Trailblazing in a Paper-like Hypertext, in The Proceedings of HyperText 98, Pittsburgh, PA, June 98, pp. 30-39.

[9] Rhodes, B. and Starner, T. The Remembrance Agent: A continuously running automated information retrieval system, in The Proceedings of PAAM '96, London, UK, April 1996, pp. 487-495

[10] Rhodes, B. The Wearable Remembrance Agent: A system for augmented memory, Personal Technologies Journal Special Issue on Wearable Computing, Personal Technologies (1997), 1:218-224.

[11] Salton, G. Automatic Text Processing. The Transformation, Analysis and Retrieval of Information by Computer. Reading, MA, Addison-Wesley.

[12] Walker, S., Robertson, S.E., Boughanem, M., Jones, G.J.F., Sparck-Jones, K., Okapi at TREC-6, in The Sixth Text Retrieval Conference (TREC-6). Edited by D.K. Harman, Gaithersburg, MD: NIST, 1998.

[13] Zellweger, P., Chang, B., Mackinlay, J. Fluid Links for Informed and Incremental Link Transitions, in The Proceedings of HyperText 98, Pittsburgh, PA, June 98, pp. 50-57.